

An Adaptive Neural Network Controller Based on PSO and Gradient Descent Method for PMSM Speed Drive

Zaineb Frijet, Ali Zribi, Mohamed Chtourou

Control & Energy Management Lab (CEM LAB), National School of Engineering of Sfax, University of Sfax, Tunisia

Article Info

Article history:

Received Nov 24, 2017

Revised May 16, 2018

Accepted Aug 6, 2018

Keyword:

Adaptive neural network
controller
Gradient descent
Particle swarm optimization
Permanent magnet synchronous
motor
Speed controller

ABSTRACT

In this paper, based on the combination of particle swarm optimization (PSO) algorithm and neural network (NN), a new adaptive speed control method for a permanent magnet synchronous motor (PMSM) is proposed. Firstly, PSO algorithm is adopted to get the best set of weights of neural network controller (NNC) for accelerating the convergent speed and preventing the problems of trapping in local minimum. Then, to achieve high-performance speed tracking despite of the existence of varying parameters in the control system, gradient descent method is used to adjust the NNC parameters. The stability of the proposed controller is analyzed and guaranteed from Lyapunov theorem. The robustness and good dynamic performance of the proposed adaptive neural network speed control scheme are verified through computer simulations.

Copyright © 2018 Institute of Advanced Engineering and Science.
All rights reserved.

Corresponding Author:

Ali Zribi,
Control & Energy Management Lab (CEM LAB), National School of Engineering of Sfax,
University of Sfax, Tunisia.
Email: ali_zribi@yahoo.fr

1. INTRODUCTION

In recent years, mechanisms driven by a permanent magnet synchronous motor (PMSM) have been widely used thanks to its advantageous merits of performances, cost and reliability. However, by reason of the continuous variation of parameters, the nonlinearity of system and the inaccessibility of some states and output for measurements, improving the PMSM performances control may become challenging [1-5]. To surpass the above problems, many artificial intelligence techniques have been proposed. Neural network (NN), due to its simple conception and easy implementation, has been widely used in industrial drive applications of PMSM [6, 7]. Recently, to improve the control precision, Chao et al. proposed a new strategy of speed control for PMSM where a combination of adaptive back propagation neural network (BPNN) and PID was adopted [8]. The convergence of the proposed control scheme was proved using the Lyapunov Stability Analysis. In order to improve the robustness performance of a PMSM control, an enhanced robust fractional order proportional-plus-integral (ERFOPI) controller is presented [9]. The proposed control law is acted on a fractional order implement function (FOIF) of tracking error. Firstly, to get the parameters of the proposed controller, different tuning rules were adopted to generate the ERFOPI parameters. Secondly, to take changes of the process parameters into consideration NN was used to adjust the controller parameters. Vikas et al. in [10] proposed a NN based PID (NNPID) like controller which is tuned when the controller is operating in an on line mode for high performance PMSM position control. In the first hand, the gains of the NNPID are initialized according to a new training algorithm based on the least square solution. Then, a sequential training algorithm is used to adjust on line the controller parameters. By estimating disturbances, Reference [11] shows a simple method of improving control performance for the speed of PMSM. They

introduced a controller builded by proportional feedback controller and feedforward compensation based on a radial basis function network disturbance observer. The proposed controller has shown good properties in disturbance rejection and in parameter variations. In order to meet higher performance specifications for a PMSM, an adaptive modified recurrent Legendre neural network control system is proposed in [12]. It consists of a modified recurrent Legendre NN control with adaptation law and a compensated control with estimation law, derived by using the Lyapunov stability theorem.

Despite control methods based on NN have a good robustness against parameter variations and unknown external disturbances, they are confronted by many difficulties. For example, NN parameters training is time-consuming and easily falls into local minimum which making them entirely dependent on initial weights [13]. PSO algorithm, as a new global optimization algorithm, has the ability to overcome these problems by accelerating the rate of convergence and preventing weights trapping into local optima. Recently, various controller design methods based on PSO and artificial neural network (ANN) have been developed [14-16]. As a new training method for feedforward neural networks (FNNs), an hybrid of PSO and Gravitational Search Algorithm (GSA) is proposed in [17]. It was employed to reduce the problems of trapping in local minima and to accelerate the convergence rate of the learning algorithms. In [18], Yaghini et al. have combined the ability of metaheuristics and greedy gradient based algorithms to obtain a hybrid improved opposition based PSO and a BP algorithm with the momentum term. Effectiveness of the proposed method is compared with several other famous ANN training algorithms on the various benchmark problems.

The purpose of this study is to develop an adaptive NN controller to improve PMSM speed control. In one hand, in order to avoid tapping in local minimum and reducing the training time, PSO algorithm is adopted for selecting the optimal solution of initial weights of the neural network controller (NNC). Then, to adapt the initially uncertain and varying parameters in the control system, the gradient descent method is used to optimize the weights where the convergence of the NN controller is guaranteed from Lyapunov theorem.

This paper is organized as follows. The PMSM model is described in section 2. Section 3 presents the NN speed controller parameters tuning using PSO algorithm and gradient descent method. In section 4, the stability analysis method is developed. The simulation results are presented in Section 5 and the conclusion is presented in Section 6.

2. BASIC PMSM MODEL

The machine model of a PMSM can be described in the rotor rotating reference frame as follows [19]:

$$\begin{cases} V_d = R_s I_d + L_s \frac{dI_d}{dt} - \omega_r L_q I_q \\ V_q = R_s I_q + L_q \frac{dI_q}{dt} + \omega_r L_d I_d + \omega_r \varphi \end{cases} \quad (1)$$

where V_d and V_q are d-q axis stator voltages, I_d and I_q are d-q axis stator currents, L_d and L_q are d-q axis stator inductances, ω_r is the electrical rotor speed and φ is the flux linkage. The electric torque can be expressed as:

$$T_e = p \left[(L_d - L_q) I_d I_q + \varphi I_q \right] \quad (2)$$

and the equation of the motor dynamics is:

$$J \frac{d\omega_r}{dt} = T_e - T_L - f \omega_r \quad (3)$$

J is moment of inertia. T_e , T_L are electromagnetic and load torques. f is friction coefficient. p is number of pole pairs. Assume that $L_d = L_q$ in a non-salient pole machine (surface mounted) PMSM, the model will be even simpler as indicated by the following equations [20]:

$$\begin{cases} \frac{dI_d}{dt} = -\frac{R_s}{L_q} I_q + p\omega_r I_q + \frac{1}{L_q} V_d \\ \frac{dI_q}{dt} = -\frac{R_s}{L_q} I_d - p\omega_r I_d - \frac{\varphi}{L_q} p\omega_r + \frac{1}{L_q} V_q \\ \frac{d\omega_r}{dt} = \frac{3p}{2J} \varphi I_q - \frac{1}{L_q J} T_L - \frac{B}{J} \omega_r \\ T_e = 1.5 p \varphi I_q \end{cases} \quad (4)$$

According to the system of equations (4), the speed control is achieved by controlling the current of two axes (I_d and I_q). The machine, speed feedback, speed and current controllers, and inverter constitute the PMSM drive system as shown in Figure 1.

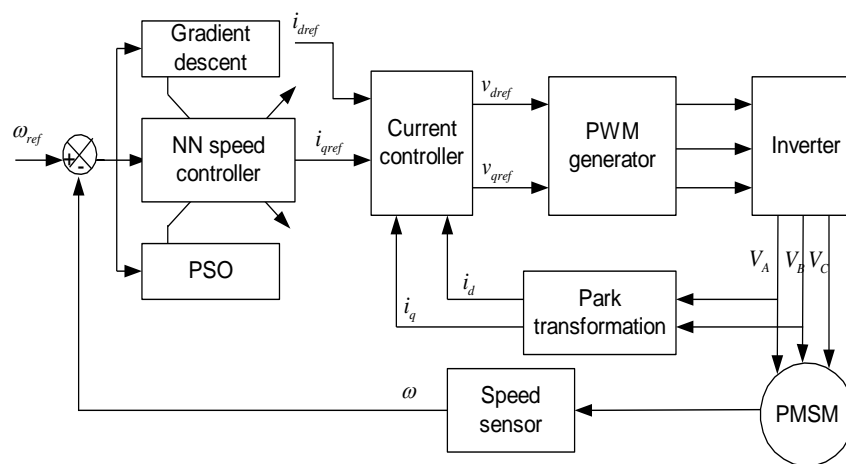


Figure 1. Block diagram of PMSM drive

3. CONTROL STRATEGY

3.1. Adaptive NN Speed Controller

The proposed speed controller for the PMSM is built based on an adaptive NN. In order to avoid trapping in a local minimum and accelerate training convergence, PSO algorithm is adopted firstly to select initial weights. The goal of PSO algorithm is to get the optimal set of weights (particles' position) by adjusting the trajectories of the particle based on the best positions. Then, to achieve high-performance speed tracking despite of the existence varying parameters in the control system, gradient descent method is used to adjust the network parameters.

The proposed method for determination the NN speed controller parameters comprises the following steps.

- Step 1: Input the object value of controlled system into the controller.
- Step 2: Search optimal NN speed controller parameters by PSO algorithm.
- Step 3: Use the obtained controller to control the PMSM.
- Step 4: Feedback the output of the PMSM.
- Step 5: Adjust parameters of NN speed controller by the gradient descent method.
- Step 6: If the speed error is small enough, then stop else go to Step 4.

3.2. PSO for feedforward NN training

The training system of the NN speed controller is determined as described in Figure 2. In this structure, the NN speed controller precedes the PMSM model and receives as input the system reference with the past system outputs and past inputs. The error signal, the difference between the reference signal and the system output, will be used by PSO algorithm to train the network.

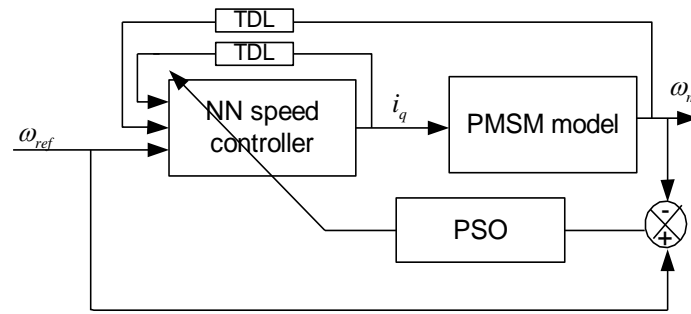


Figure 2. Control loop implemented during the PSO of the NN speed controller

The training process of NN is usually complicated and high dimensional. The PSO as an evolutionary algorithm could be used in neural network training [22]. It is a population (swarm) based optimization tool. Every single solution (called a particle) “flies” over the solution space in search for the optimal solution. In each generation, each particle is updated based on 2 special particles: P_b is the personal best solution of each particle, and P_g is the global best solution by any particle in the swarm (population). The performance of each particle is measured using a fitness function that varies depending on the optimization problem [23, 24]. These particles velocity and position are updated through the following equations:

$$v_{id}(k+1) = wv_{id}(k) + C_1 r_1 (p_{id}(k) - x_{id}(k)) + C_2 r_2 (p_{gd}(k) - x_{id}(k)) \quad (5)$$

$$x_{id}(k+1) = x_{id}(k) + v_{id}(k+1) \quad 1 \leq d \leq D \quad (6)$$

Where, D is the dimension for a searching space, $X_i = (x_{i1}, x_{i2}, \dots, x_{iD})$ represents the position of each particle, $V_i = (v_{i1}, v_{i2}, \dots, v_{iD})$ is the velocity of the i th particle, $P_{ib} = (p_{i1}, p_{i2}, \dots, p_{iD})$ is the best position encountered by i th particle, P_g shows the best position found by any member in the entire swarm population $P_g = (p_{g1}, p_{g2}, \dots, p_{gD})$, k is iteration counter; C_1 , C_2 are acceleration coefficients and r_1 , r_2 are two similar random numbers in $[0, 1]$.

w is called the inertia factor. It reduces during a run from 1 to near 0 in each generation which facilitates a balance in the exploration and exploitation of the search space, it is determined as follows:

$$w = w_{\max} - \frac{w_{\max} - w_{\min}}{iter_{\max}} \times iter \quad (7)$$

where $iter_{\max}$ is the maximum number of iterations, and $iter$ is the current number of iteration.

PSOBP Algorithm

In this study, PSO is used to train a NN to obtain an optimum network model and to improve the performance of the NN. During the training phase, the mean squared error (MSE) is used to calculate the fitness value.

$$MSE = \frac{1}{N} \sum_{i=1}^N e_i \quad (8)$$

where e_i is the error between desired and obtained outputs after presenting the i th datum to the network, and N is the number of data in the training dataset.

The procedure of the PSOBP can be described as follows where a particles' position represents the values for the networks' weights and biases.

Step 1: Initialize the positions and velocities of a group of particles randomly in the range of $[0, 1]$.

Step2: Evaluate each initialized particles' fitness value and P_b is set as the positions of the current particles, while P_g is set as the best position of the initialized particles.

Step3: Change the velocity of the particle according to (5) and update particle position by adding the calculated velocity value to the current position value according to (6).

Step 4: Compare the current value of the MSE with the particles' previous best value (P_{ib}). If the current fitness value is less, then updating P_{ib} .

Step 5: Finding the current global minimum of MSE and compare it with the previous global minimum (P_g). If the current global minimum is better than P_g , then updating P_g .

Step 6: If maximum number of iterations is reached or the fitness values are met, stop the iteration, and the positions of particles are the optimal best solution. Otherwise, the process is repeated from step3.

Step 7: Taking the weights and biases values which optimized by PSO as the initial parameters, the BP network makes autonomous learning.

3.3. NN Controller Parameters Adjustment

The NN speed controller weights, optimized by the PSO, will be used to set initial weights in the controller. When the system work situation changes degradation in the control performances can be accrued. To get better control effect and close quickly to control object values, at each control cycle, weights must be adjusted according to the error. The gradient descent method can be used to minimize the system error between the output of the PMSM and the desired output. Objective function is defined as follows:

$$E(k) = \frac{1}{2} (\omega_{ref}(k) - \omega(k))^2 \quad (9)$$

where ω is the actual output and ω_{ref} is the desired output.

The weights updating is calculated using the gradient method, the expressions (10) and (11) define the formula.

$$w_{ij}(k+1) = w_{ij}(k) - \eta \frac{\partial E(k)}{\partial w_{ij}} \quad (10)$$

$$w_j(k+1) = w_j(k) - \eta \frac{\partial E(k)}{\partial w_j} \quad (11)$$

Where η is the learning rate and w_{ij} are the weights from the input layer to the hidden layer and w_j are the weights from the hidden layer to the output layer.

In each sampling period, to adapt varying parameters in the control system all weights characterized the NN controller are updated according to the errors of closed loop system.

As the hidden and output neuron functions were defined by the logistic sigmoid function, the quantities in (10) and (11) are respectively calculated by the expressions (12) and (13).

$$\frac{\partial E(k)}{\partial w_{ij}(k)} = \frac{\partial E(k)}{\partial \omega(k)} \frac{\partial \omega(k)}{\partial i_q(k)} \frac{\partial i_q(k)}{\partial w_{ij}(k)} = -E(k) \frac{\partial \omega(k)}{\partial i_q(k)} i_q(k)(1-i_q(k))w_j O_j (1-O_j) O_i \quad (12)$$

$$\frac{\partial E(k)}{\partial w_j(k)} = \frac{\partial E(k)}{\partial \omega(k)} \frac{\partial \omega(k)}{\partial i_q(k)} \frac{\partial i_q(k)}{\partial w_j(k)} = -E(k) \frac{\partial \omega(k)}{\partial i_q(k)} i_q(k)(1-i_q(k)) O_j \quad (13)$$

O_j is the output of j^{th} neuron in the hidden layer.

The value of the partial derivative $\frac{\partial \omega}{\partial i_q}$ can be replaced by $\text{sign} \left[\frac{\partial \omega}{\partial i_q} \right] = 1(-1)$.

4. STABILITY ANALYSIS

The on-line training algorithms for the speed controller calls for a proper choice of learning rate η . Let Lyapunov function be

$$V(k) = \frac{1}{2} e^2(k) \quad (14)$$

where

$$e(k) = \omega_{ref}(k) - \omega(k) \quad (15)$$

The change of Lyapunov function is

$$\Delta V(k) = V(k+1) - V(k) = \frac{1}{2} (e^2(k+1) - e^2(k)) = \Delta e(k) \left[e(k) + \frac{1}{2} \Delta e(k) \right] \quad (16)$$

The error difference can be represented as

$$e(k+1) = e(k) + \Delta e(k) = e(k) + \left[\frac{\partial e(k)}{\partial W(k)} \right]^T \Delta W(k) \quad (17)$$

$W(k)$ is the networks weight vectors. The expression of $\Delta W(k)$ is given by

$$\Delta W(k) = -\eta \frac{\partial e(k)}{\partial W(k)} = -\eta \frac{\partial e(k)}{\partial \omega(k)} \frac{\partial \omega(k)}{\partial i_q(k)} \frac{\partial i_q(k)}{\partial W(k)} = \eta e(k) \omega_{i_q}(k) \frac{\partial i_q(k)}{\partial W(k)} \quad (18)$$

Based on Equation (17) and Equation (18) we can obtained

$$\Delta e(k) = \left[\frac{\partial e(k)}{\partial W(k)} \right]^T \eta e(k) \omega_{i_q}(k) \frac{\partial i_q(k)}{\partial W(k)} \quad (19)$$

Substituting the expression of $\Delta e(k)$ in $\Delta V(k)$, we have

$$\begin{aligned} \Delta V(k) &= \left[\frac{\partial e(k)}{\partial W(k)} \right]^T \eta e(k) \omega_{i_q}(k) \frac{\partial i_q(k)}{\partial W(k)} \\ &\quad \times \left\{ e(k) + \frac{1}{2} \left[\frac{\partial e(k)}{\partial W(k)} \right]^T \eta e(k) \omega_{i_q}(k) \frac{\partial i_q(k)}{\partial W(k)} \right\} \end{aligned} \quad (20)$$

Since $\frac{\partial e(k)}{\partial W(k)} = -\omega_{i_q}(k) \frac{\partial \omega(k)}{\partial W(k)}$, substituting this in Equation (20), we have,

$$\Delta V(k) = -\eta e^2(k) (\omega_{i_q}(k))^2 \left[\frac{\partial i_q(k)}{\partial W(k)} \right]^T \frac{\partial i_q(k)}{\partial W(k)} + \frac{1}{2} \eta^2 e^2(k) (\omega_{i_q}(k))^4 \left\{ \left[\frac{\partial i_q(k)}{\partial W(k)} \right]^T \frac{\partial i_q(k)}{\partial W(k)} \right\}^2 \quad (21)$$

In our case, the value of the partial derivative $\omega_{i_q} = \frac{\partial \omega}{\partial i_q}$ is replaced by $\text{sign} \left[\frac{\partial \omega}{\partial i_q} \right] = 1(-1)$. We

obtain

$$\begin{aligned}
\Delta V(k) &= -\eta e^2(k) \left\| \frac{\partial i_q(k)}{\partial W(k)} \right\|^2 + \frac{1}{2} \eta^2 e^2(k) \left\| \frac{\partial i_q(k)}{\partial w(k)} \right\|^4 \\
&= -\frac{1}{2} \left\| \frac{\partial i_q(k)}{\partial W(k)} \right\|^2 \eta \left(2 - \eta \left\| \frac{\partial i_q(k)}{\partial W(k)} \right\|^2 \right) e^2(k)
\end{aligned} \tag{22}$$

According to the Lyapunov Theorem [25], the closed loop system is stable if

$$\Delta V(k) \leq 0 \tag{23}$$

So

$$2 - \eta \left\| \frac{\partial i_q(k)}{\partial W(k)} \right\|^2 \geq 0 \tag{24}$$

Finally, when we define the matrix norm $\|\cdot\|_2$ by

$$\|M\|_2 = \sqrt{\max(M^T M)} \tag{25}$$

the theorem result is established. The stability condition $\Delta V(k) \leq 0$ is satisfied only if

$$\eta \leq \frac{2}{\beta_{\max}^2(k)} \quad \text{where} \quad \begin{cases} \beta(k) = \frac{\partial i_{qr}(k)}{\partial W(k)} \\ \text{and} \\ \beta_{\max}(k) = \|\beta(k)\|_2 \end{cases} \tag{26}$$

5. SIMULATIONS RESULTS

To verify the effectiveness of the proposed high speed PMSM control algorithm, a digital simulation based on the Matlab/Simulink software package has been carried out. The PMSM characteristics are given in Table 1.

Table 1. Machine Parameters

Parameter	Value
Rated voltage	220/380 V
Stator resistance (R_s)	1.4 Ω
Inductance (L_d , L_q)	6.6 mH
Magnetic flux constant (Φ)	0.1546 Wb
Motor inertia (J)	0.00176 N.m.s ² /rad
Friction coefficient (f)	0.000388 N.m. s/rad
Pole pairs numbers (p)	3

The function of PSO in NN speed controller is to get the best set of weights (particles' positions) where several particles are moving to have the best solution. The NN is trained using the training system described in section 3 and Figure 2. In our proposed method, the PSO is adopted to select initial weights. The dimension of each particle is the total number of weights and biases. During the process of weights initiation by PSO, numbers of populations is specified as 21, and iteration number is specified as 100.

In the training phase, the setpoint sequence consisting of pulses of random amplitude in the range $[0; 300\text{rad/sec}]$, and with a duration of 20 sampling periods. The maximum amplitudes are chosen so as to fully explore the desired speed amplitude range.

To illustrate the advantages of our controller, experiments were executed using two speed controllers such as: the proposed controller and a conventional adaptive NN controller with a random initialization. Figure 3 shows PMSM model response using NNC optimized by PSO for random setpoint. ((dash-dotted) setpoint ω_{ref} ; (solid) PMSM model response ω_m).

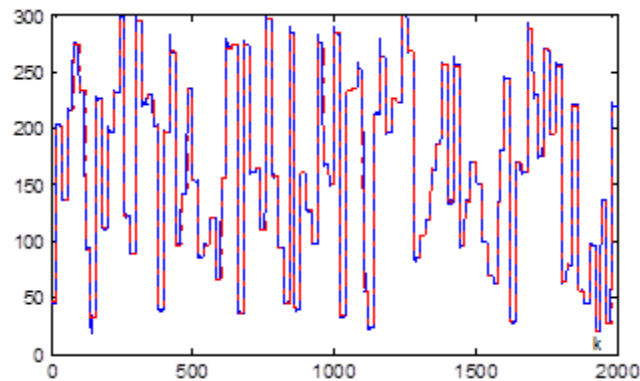


Figure 3. PMSM model response using NNC optimized by PSO for random setpoint. ((dash-dotted) setpoint ω_{ref} ; (solid) PMSM model response ω_m)

Figure 4 shows the speed tracking performance test of both the controllers. It is clear that the proposed controller has the ability to keep good control performance in the entire process where the overshoot, settling time and rise time are reduced.

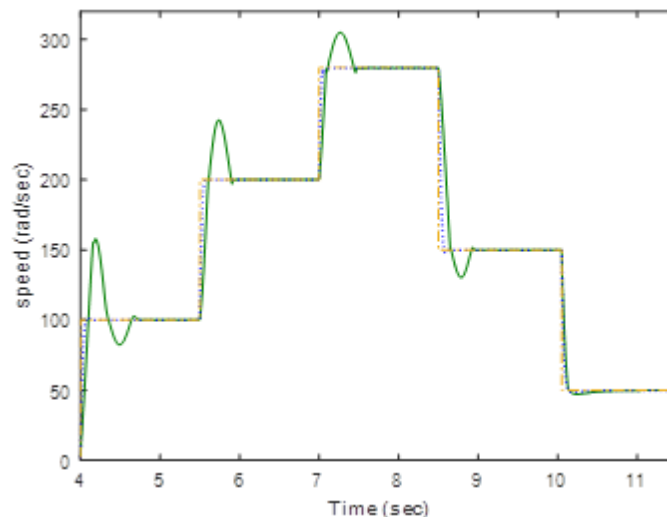


Figure 4. Speed tracking responses. ((dash-dotted) setpoint; (solid) with adaptive NN controller; (dotted) with the proposed controller)

Figure 5 shows the stator current q-components. Each change in the speed reference is accompanied by a peak in their shapes. It can be observed that at every changing operating condition the proposed controller demonstrates a fast current control response.

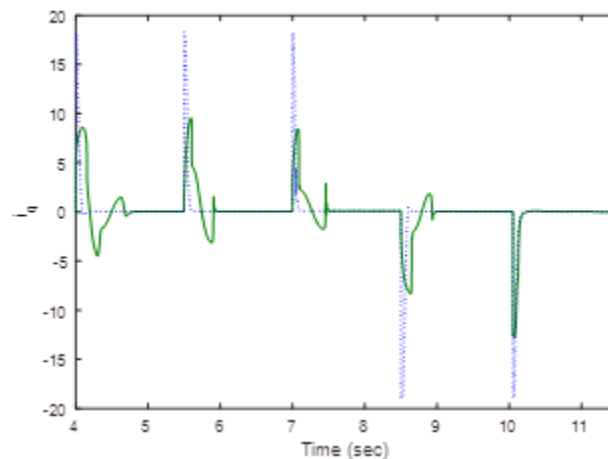


Figure 5. Stator current q-axis-components. ((solid) with adaptive NN controller; (dotted) with the proposed controller)

The purpose of a second test is to evaluate the disturbances rejection effectiveness in presence of step load torque disturbance. The system is tested with value of 5 Nm load at 5.0s with 100rad/sec speed reference. From Figure 6, it is obvious that the proposed controller drop the speed from 100rad/sec to 90rad/sec with short settling time. While, adaptive NN controller with a random initialization drop the speed from 100rad/sec to 63rad/sec with largest settling time. The proposed speed controller produce significant improvement control performance compare to the adaptive NN controller such as reduce overshoot, settling time and rise time speed response.

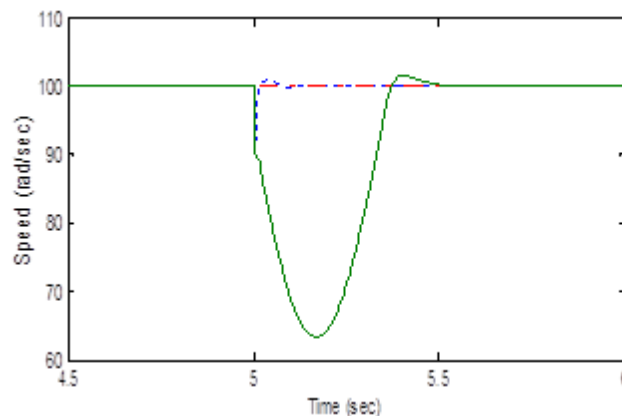


Figure 6. Disturbance rejection responses. ((dash-dotted) setpoint; (solid) with adaptive NN controller; (dotted) with the proposed controller)

The system robustness against the motor inertia changes is given in Figure 7. As can be seen, the increment of the moment of inertia does not impose any significant effect on the performance of the proposed controller technique but only affects the rise time. Furthermore, when carefully study Figure 7 according to the settling time and overshoot, the best performance belongs to the proposed controller. This means that the proposed controller is insensitive to parametric variations and a robust tracking performance is achieved in presence of the uncertain parameters.

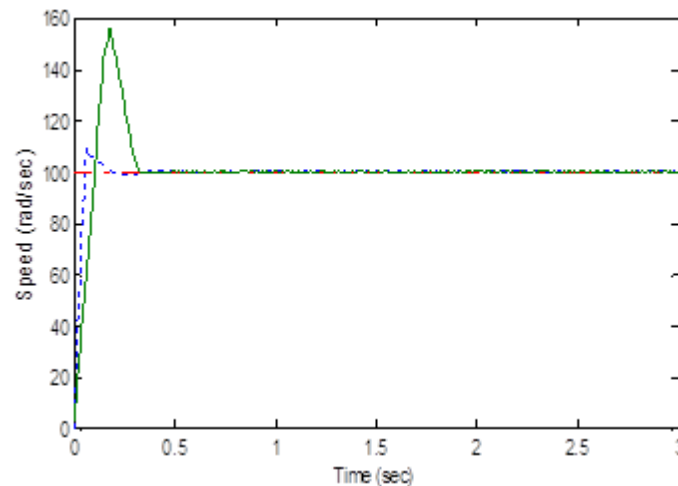


Figure 7. Responses of the PMSM with variation 100% in the moment of inertia((dash-dotted) setpoint; (solid) with adaptive NN controller; (dotted) with the proposed controller)

6. CONCLUSION

In this paper, we have proposed a novel approach of speed control for PMSM combining PSO and adaptive NN. PSO algorithm was adopted to select initial weights and bias, presented as particles position, which were updated using movement and velocity update equations to avoid the weight and biases being trapped in local minima. Then, in each sampling period, to adapt the initially uncertain and varying parameters in the control system the gradient descent method was used to update the weights of controller.

REFERENCES

- [1] E. Cetin E and U. Qguz, " A hybrid controller for the speed control of a permanent magnet synchronous motor drive," *Cont. Eng. Pract.*, vol. 16, no. 3, pp. 260-270, 2008.
- [2] M. Karabacak and H. I. Eskikurt, " Design, modelling and simulation of a new nonlinear and full adaptive backstepping speed tracking controller for uncertain PMSM," *Applied Mathematical Modelling*, vol. 36, pp. 5199-5213, 2013.
- [3] T. J. Ren and T. C. Chen, " Robust speed-controlled induction motor drive based on recurrent neural network," *Applied Mathematical Modelling*, vol. 76, pp. 1064-1074, 2006.
- [4] M. Moujahed *et al.*, " Extended Kalman Filter for Sensorless Fault Tolerant Control of PMSM with Stator Resistance Estimation," *IJPEDS*, vol. 9, pp. 579-590, 2018.
- [5] C. Lin and R.Wu, " Adaptive Recurrent Network Network Uncertainty Observer Based Integral Backstepping Control for a PMSM Drive System," *IJPEDS*, vol. 2, pp. 127-137, 2012.
- [6] S. Richa, *et al.*, " An adaptive PID like controller using mix locally recurrent neural network for robotic manipulator with variable payload," *ISA Transactions*, vol. 62, pp. 258-267, 2016.
- [7] N. Muhd *et al.*, " Speed Control of Permanent Magnet Synchronous Motor Using FOC Neural Network," the series Lecture Notes in Electrical Engineering, pp. 295-203, 1996. (Book style)
- [8] N. Muhd, *et al.*, " Speed Control of Permanent Magnet Synchronous Motor Using FOC Neural Network," the series Lecture Notes in Electrical Engineering, pp. 295-203, 1996. (Book style)
- [9] B. Zhang and Y.Pi, " Enhanced robust fractional order proportional-plus-integral controller based on neural network for velocity control of permanent magnet synchronous motor," *ISA Transactions*, vol. 52, pp. 510-516, 2013.
- [10] K. Vikas, *et al.*, " ANN based self tuned PID like adaptive controller design for high performance PMSM position control," *Expert Systems with Applications*, vol. 41, pp. 7995-8002, 2014.
- [11] L. Juan, *et al.*, " . Adaptive speed control of a PMSM servo system using an RBFN disturbance observer," *Transactions of the Institute of Measurement and Control*, vol. 34, pp. 615-621, 2011.
- [12] H. L. Chih. Novel application of continuously variable transmission system using composite recurrent Laguerre orthogonal polynomials modified PSO NN control system," *ISA Transactions*, vol. 64, pp. 405-416, 2016.
- [13] G. Marco and T. Alberto. On the problem of local minima in back-propagation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, pp. 76-86, 1992.
- [14] A. Ajith, *et al.*, " Swarm intelligence: foundations, perspectives and applications," *Studies in computational intelligence*, vol. 26, pp. 3-25, 2006.

- [15] H. L. Chih. Novel adaptive modified recurrent Legendre neural network control for a PMSM servo-driven electric scooter with V-belt continuously variable transmission system dynamics," *Transactions of the Institute of Measurement and Control*, vol. 37, pp. 1181-1196, 2015.
- [16] J. M. Mohammad, B. Mahdis B. An online optimal linear state feedback controller based on MLS approximations and a novel straightforward PSO algorithm," *Transactions of the Institute of Measurement and Control*, vol. 37, pp. 1132-1142, 2014.
- [17] A.M. Seyed, *et al.*, " Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm," *Applied Mathematics and Computation*, vol. 218, pp. 11125-11132, 2012.
- [18] M. Yaghini, *et al.*, "A hybrid algorithm for artificial neural network training," *Engineering Applications of Artificial Intelligence*, vol. 26, pp. 293-301, 2013.
- [19] A. Nait Seghir and M.S. Boucherit, " Artificiel neural network to improve speed control of permanent magnet synchronous motor," In: *International Conference on Electric Power Systems, High Voltages, Electric Machines, Tenerife, Spain*, pp. 16-18, 2006.
- [20] A.A. Aishwara, *et al.*, " Speed Control of PMSM Using Disturbance Observer," In: *Conference International Federation of automatic control*, pp. 308-313, 2016.
- [21] R.Z. Jing, *et al.*, "A hybrid particle swarm optimization–back-propagation algorithm for feedforward neural network training," *Applied Mathematics and Computation*, vol. 185, pp. 1026-1037, 2007.
- [22] C. DeBao and Z. ChunXia, " Particle swarm optimization with adaptive population size and its application," *Applied Soft Computing*, vol. 9, pp. 939-948, 2009.
- [23] K. Stefan, *et al.*, " Adaptive stochastic gradient descent optimization for image registration," *International Journal of Computer Vision*, vol. 81, pp. 227-239, 2009.
- [24] S. W. Shneen *et al.*, " Advanced Optimal PSO, Fuzzy and PI Controller with PMSM and WTGS at 5Hz Side of Generation and 50Hz Side of Grid," *IJPEDS*, vol. 7, pp. 173-192, 2016.
- [25] J. Peng and R. Dubay, " Identification and adaptive neural network control of a DC motor system with dead-zone characteristics," *ISA Transactions*, vol. 50, pp. 588-599, 2011.